Bag-of-Words based Natural Language Inference

Haoxue Li (hl3664), Xiaocheng Li (xl3119), Diwen Lu (dl3209), Gaomin Wu (gw1107) *by last name*

Abstract— **In this project, we train a Bag-Of-Words encoder to tackle the multi-label task using the Stanford Natural Language Inference (SNLI) and Multi-Genre Natural Language Inference (MultiNLI) Dataset. Our source code is publicly available at `https://github.com/DiwenLu/BOW_NLI`.**

INTRODUCTION

We in this project build a Bag-Of-Words encoder to identify the logic between a ***premise*** and a ***hypothesis***. Given two pieces of texts, a *premise* and a *hypothesis*, we determine whether the premise **entails**, **contradicts**, or is **neutral** to the hypothesis. This is a three-class classification problem.

- ***Premise***: A man inspects the uniform of a figure in some East Asian country.
- ***Hypothesis***: The man is sleeping.
  This is a contradiction, since the man cannot both be sleeping and inspecting the uniform.

- ***Premise***: A soccer game with multiple males playing.
- ***Hypothesis***: Some men are playing a sport.
  This is an entailment, because if the former is true, the latter must be true.

- ***Premise***: An older and younger man smiling.
- ***Hypothesis***: Two men are smiling and laughing at the cats playing on the floor.
  This is neutral, because the premise does not contain any information about cats that the hypothesis posits.

In the Section 1., we show the description of the data set we used. We then show the model construction in Section 2. We first build two classifiers, one Logistic Regression and one Neural Network, and train them on SNLI. We then perform hyper-parameter tuning for these two models on vocabulary size, embedding dimension, and different interaction methods for encoded hypothesis and premise. Then, we choose two trained model with best hyper-parameter setting, one each for LR and NN based classifier, and evaluate them on the validation data set of SNLI with detailed training and validation accuracy and loss, and perform error analysis on it. In Section 3.2, we evaluate the trained model with the provided MultiNLI data for each genre. In Section 3.3, we picked the best trained model over the LR and NN and trained it further on training data of each MultiNLI genre. After fine-tuning, the validation accuracy does increase. In section 3.4, we use a frozen pre-trained word embedding to implement the model. In particular, we use two pre-trained word embedding ways: one is to freeze all pre-trained word embedding, the other is to learn unknown token and freeze other pre-trained embedding. In section 3.5, we take 10 most similar pair of words in embedding in 3.1. We compare the distance in trained embedding in 3.1 with the new distance in frozen pre-trained embedding and analyze the reason behind it.

1. Dataset

SNLI refers to Stanford Natural Language Inference, which collected 570k human-written English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral. In our project, we are provided the pre-tokenized training (100,000 examples) and validation (1,000 examples) sets of the SNLI in a .tsv format.

MultiNLI refers to Multi-Genre Natural Language Inference, which is a crowd-sourced collection of 433k sentence pairs annotated with textual entailment information. The corpus is modeled on the SNLI corpus, but differs in that covers genres: telephone, fiction, slate, government, travel. In our project, we are provided the pre-tokenized training (20,000 examples) and validation (5,000 examples) in a .tsv format.

2. Modeling

We use a BoW encoder to map each string of text (hypothesis and premise) to a fixed-dimension vector representation. At this point, we have one vector representation for hypothesis and one for premise. Then we can interact the two representations by 1) take the sum 2) take Hadamard product and 3) concatenate them. After we complete embedding and combine representation of hypothesis and premise, we build two models, logistic regression and a fully connected neural network with two hidden layers.

3.1 Training on SNLI

In training the model on SNLI, in default, our learning rate is 0.01, batch size is 1024, max sentence length is 20, and epoch number is 10. And our NN model has 90 neurons in both the 1st hidden and the 2nd hidden layer. After we implemented two baseline models respectively for LR and NN by taking summation of encoded hypothesis and premise, using vocabulary size at 10,000 and embedding dimension at 50, we started to fine tune 2 sets of hyperparameters. We took vocabulary size from [10,000, 50,000, 100,000] and embed dimension from [50, 100, 200]. We also experimented with taking the sum, hadamard product, and concatenating two encoded sentences. In the end, we had 54 models to train with, 27 for LR, and 27 for NN. Here we chose classification accuracy as the validation metric.

The following shows the validation accuracy per model per epoch. The validation accuracy per model is taken to be the max among its 10 epochs.

Vocab	Embed	Interact	NN_val	LR_val
10k	50	concat	66.13	62.61
10k	50	hadamard	64.80	**66.52**
10k	50	sum	59.82	57.51
10k	100	concat	65.64	62.74
10k	100	hadamard	60.55	63.59
10k	100	sum	59.63	56.79
10k	200	concat	65.96	61.84
10k	200	hadamard	61.51	63.93
10k	200	sum	59.04	56.57
50k	50	concat	**66.43**	62.28
50k	50	hadamard	61.58	64.34
50k	50	sum	59.92	57.61
50k	100	concat	66.08	62.22
50k	100	hadamard	61.51	65.23
50k	100	sum	59.30	57.32
50k	200	concat	65.49	61.61
50k	200	hadamard	61.66	63.45
50k	200	sum	60.70	56.62
100k	50	concat	65.89	62.49
100k	50	hadamard	63.07	64.22
100k	50	sum	58.24	56.65
100k	100	concat	65.85	62.57
100k	100	hadamard	62.23	62.61
100k	100	sum	59.35	57.50
100k	200	concat	65.92	62.37
100k	200	hadamard	64.50	61.62
100k	200	sum	59.99	56.64

TABLE I: Validation accuracy of LR and NN model with all hyper-parameter combinations

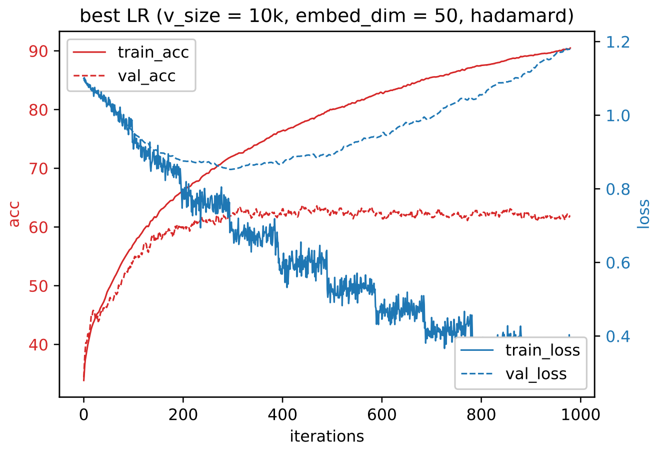

Fig. 1: Learning curve with train & validation loss, train & validation accuracy (on the whole training data set) of the best LR classifier

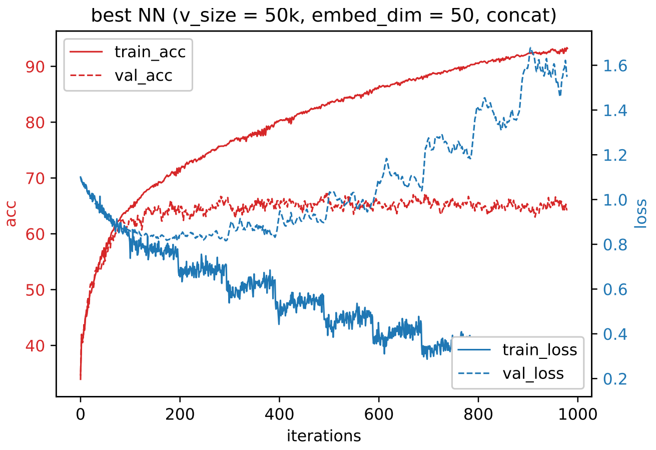

Fig. 2: Learning curve with train & validation loss, train & validation accuracy (on the whole training data set) of the best NN classifier

Hyperparameter Implication

Interaction: Interactions refer to how we combine two embedded representations of hypothesis and premise. Concatenation means we let the model help us to find the relation between the hypothesis and premise embedding. Hadamard product means we want to multiply the corresponding component together in hypothesis and premise, so the similar words and brought closer and dissimilar words are brought further. Adding is simply another way to describe the relation between corresponding words in hypothesis and premise and is less effective than hadamard product.

Vocabulary Size: The more vocabulary we have, the less uncommon words we would see in our training and validation data set, and the more information we have about a sentence.

Embedding Dimension: Embedding is used to decrease the dimension of our sentence vector. The lower the embedding dimension, the more the information about a sentence is integrated, and the less each sentence can be identified from the other, so the lower training and validation accuracy we should see. However, in our tuning, the embedding dimension at 50 and 100 doesn't make huge difference.

Performance Analysis of Validation Dataset

In this subsection, we chose two best models for NN and LR and evaluated their classification performance on SNLI Validation data set. LR (vocab_size = 10k, embed_dimension = 50, interaction = hadamard) reached max validation accuracy after the 4th epoch. NN (vocab_size = 50k, embed_dimension = 50, interaction = concat) reached max validation accuracy after the 7th epoch. SNLI validataion data set has 1000 data points in total. Learning curve of two best models are shown in Figure1 and Figure2.It's obvious that the model tends to overfit after a few epochs.

Interestingly, we find hadamard is always the best interaction method in LR model whereas concatenation works best for NN model. Summation ys the worst choice for both models. Additionally, within summation, NN always outperforms LR, but hadamard works better for LR compared to NN for most of time. However, based on our choice of hyperparameters, it seems not obvious how vocabulary size and embedding dimension would impact classification accuracy.

Error Analysis of Validation Dataset

In this subsection, we choose the best LR model (Voc_size=10k, embed_dim = 50, interaction = hadamard), and then examine individual cases where model give correct and incorrect label, and try to determine which kind of information would trick the model to make a wrong decision. By looking through the examples, it's clear that the model can identify some synonymous words such as 'women'='ladies', 'male'='man' and 'woman'='female' etc. For example:

- ***Premise***: Two ladies are sitting together at the table.
- ***Hypothesis***: Two asian women talking and having drinks at a small round table.

- ***Premise***: A male and female work together to repair something.
- ***Hypothesis***: A man and a woman repairing something.

  These two pairs are correctly predicted as entailment. They also imply that entailment would be predicted if sentence has a lots same word.

Although words used be highly similar, the model can still find differences and predicted contradiction accurately, for example:

- ***Premise***: The man is riding a sheep.
- ***Hypothesis***: A man rides a kicking bull in a bullpen.

  This is correctly predicted as contradiction.

There are still some synonymous words which the model can not identify, such as 'bike' & 'tricycle' in the following example, maybe because these synonymous words are not as frequent as woman and female etc. in the training data set.

- ***Premise***: A man is on a bike
- ***Hypothesis***: A man standing on a tricycle riding in front of a crowd.

  This is correctly predicted as contradiction, while the true label is entailment.

It also shows that when the sentence is long and words used are almost same, the model would tend to predict entailment, for example

- ***Premise***: A dog floating on a river near a metropolitan area.
- ***Hypothesis***: A person floating on a river near a metropolitan area.

- ***Premise***: The man is walking his cat.
- ***Hypothesis***: Young man walking dog.

  These two pairs are all contradiction, but the model predict them as entailment.

Through this error analysis, we can identify a possible way to improve performance: add more synonymous words sample in training data.

3.2 Evaluating on MultiNLI

The best LR model takes hadamard product of encoded hypothesis and premise, uses vocabulary size at 10k, and has embedding dimension 100. The best NN model takes concatenation of encoded hypothesis and premise, uses vocabulary size at 50k, and has embedding dimension 100. The following table shows validation accuracy of the best LR and NN model for each genre in provided MultiNLI data set.

Genre	Best LR	Best NN
fiction	41.91	42.11
travel	37.88	43.18
government	36.12	42.81
slate	37.62	40.02
telephone	40.30	45.77

TABLE II: Best LR and NN validation accuracy on all genres

The overall validation accuracy on MNLI is much worse than SNLI, since MNLI categorizes data set by genres, and it introduces many unseen words to the trained model. The valuation accuracy of Best Neutral Network (50k vocabulary size, 50 embedding, concatenation) is much better than the valuation accuracy of Best LR (10K vocabulary, 50 embedding, hadamard). And this result is consistent with what we expected. Within 5 genres, 'slate' data set always has the lowest validation accuracy no matter how we configure the model, which implies the words in slate are foreign to our trained model.

3.3 Fine-tuning on MultiNLI

We see the the best NN outperformed the best LR model in all genres. Here we continue to fine tune our best SNLI-trained NN model on training data for each MultiNLI genre (training set) and evaluate it on that genre (validation set). Here we still set learning rate to be 0.01 and used the saved best NN model in 3.2 and ran for additional 20 epochs. After we incorporate each genre's training set into the fine tuning process, we see reasonable improvement on validation accuracy.

Genre	Before	After
fiction	42.11	45.65
travel	43.18	48.75
government	42.81	50.71
slate	40.02	42.12
telephone	45.77	50.47

TABLE III: Best NN fine tuned validation accuracy on all genres

3.4 Pre-Trained Word Embedding

In section 3.4, instead of learning word embedding, we use one of the fastText word vector sets: wiki-news-300d-1M vector as our frozen pre-trained word embedding. This set contains 1 million word vectors trained on Wikipedia 2017, UMBC webbase corpus and statmt.org news data set (16B tokens).

We repeat mostly what we did in section 3.1 and 3.2. For hyper parameters, we change vocabulary size and interaction type of two sentences. From fastText word vector set, the embedding size is fixed 300. We experiment on two different vocabulary size: 50k and 100k since fastText vocabulary may have very different frequency pattern from that SNLI has. We also consider three interaction types: sum, hadamard product, and concatenation. For model selection, we choose neutral network model since it outperforms LR model in our trials.

One thing noted is that there are some unknown tokens in SNLI, so we need to map them to $< UNK >$ token in the downloaded embedding vocabulary. In the following training model part, we use two methods to approach the unknowns in SNLI. One is to use all frozen pre-trained word embedding, simply ignoring the unknowns, the other is to learn $< UNK >$ token from available tokens and use frozen pre-trained word embedding for known tokens. The following table shows the validation accuracy of NN on SNLI with both unlearned and learned unknowns.

Vocab	**Interact**	**NN_unlearned**	**NN_learned**
50k	concat	64.93	66.45
50k	hadamard	54.16	54.76
50k	sum	57.00	51.89
100k	concat	**67.50**	**67.19**
100k	hadamard	56.33	56.63
100k	sum	54.84	52.09

TABLE IV: NN (unknowns unlearned and learned) validation accuracy on SNLI

We take the best neutral network model (interaction = concat, vocab_size = 100k). We repeat what we did in section 3.2: evaluate on MultiNLI data set for each genre. The following table shows validation accuracy of the best NN model with and without learned $< UNK >$ tokens for each genre.

Genre	**Best_NN_unlearned**	**Best_NN_learned**
fiction	44.02	42.71
travel	44.40	43.89
government	44.78	42.52
slate	42.12	41.82
telephone	41.59	43.68

TABLE V: NN (unknowns unlearned and learned) validation accuracy on all genres in MultiNLI

3.5 Analyzing Learned Word Embedding

We subtract embedding for each word. To find the most similar ten pairs of words, the most important step is to define the similarity between words. We try both Euclidean distance and cosine similarity, and decide to use Euclidean distance as our representation of word similarity. After sorting word-pairs according to pair-wise distances, we create a list using NLTK *stopwords* corpus, adding up determiners and number, to eliminate the pairs which include functional words.We then subtract the pair-wise distance of these pair of words using pre-trained embedding. The results are listed in **TABLE VI**.

From **TABLE VI**, we can predict that most of the word vectors are more tightly related to its semantic representation than its syntactic representation. For example, *young* and *child* have different functions in a sentence, as *young* often serves as the adjective modifier of a noun in a sentence, while *child* is a noun. Semantically, however, they both contain the meaning of ”the early phase of life, often immature.” The same relation applies to other pairs, such as *land* and *cluster* which share a sense of collectivism in the political background, and *man* and *guys* which are often the alternative form of the other in daily speech.

Similar Words	**Dist_trained**	**Dist_pretrained**
young-child	9.59	1.81
land-cluster	9.74	2.60
man-guys	9.81	1.75
group-hat	9.82	2.30
young-casual	10.06	2.06
group-wonderful	10.08	2.15
young-grass	10.07	2.33
man-sites	10.12	2.25
young-across	10.17	2.20
area-young	10.19	1.89

TABLE VI: Most similar words in trained embedding of SNLI dataset and its representation in pre-train embedding using fastText

More specifically, by looking into the domain of etymology, we can also speculate that the embedding layer also presents semantic relation between English words in a highly abstract level. The formation of *grass* can be traced back to Proto-Indo-European token $*ghros-$(where * means that the token is speculated instead of observed), which means ”young shoot, sprout.” We then look into two sentences in the training set.

- Two large brown dogs are playing in the grass.
- A young boy is sitting on the grass.

Directly and indirectly, the embedding process automatically relates grass with tokens *young*, *boy*, *playing*, which also coincide with our speculation on the abstract semantic similarity between the two tokens.

Also we have noticed that the distances of similar words in pre-trained embedding decrease in a large extent compared with those in trained embedding. We speculate that the similarity of words are more obvious since the vocabulary size increase and the context extent of words based on FastText enlarges.